Amendments to the Claims:

This listing of claims replaces all prior versions and listings of claims in the application:

Listing of Claims:

1. (Currently amended) A method of operating a multithreaded parallel processor comprising:

directing the processor having a plurality of microengines, based on a voluntary swap specified in a context-swap instruction, to swap a currently running context, corresponding to a first thread, in a specified microengine to let another context, corresponding to a different thread that is ready to execute, execute in that microengine and cause a different context and associated program counter to be selected, with the swapped first thread automatically re-enabled to run at some subsequent context arbitration point.

2. (Currently amended) The method of claim 1 wherein ~~the~~ directing the processor comprises waking ~~wakes~~ up the swapped out context when [[a]] an additional signal ~~specified in a context-swap program instruction~~ is activated.

3. (Currently amended) The method of claim 2 wherein the additional signal is specified as a parameter in the context-swap instruction ~~in the directing~~ and specifies an occurrence of an event.

4. (Currently amended) The method of claim 3 wherein the parameter specifies "sram Swap", and the directing swaps out [[a]] the current context and wakes it up when the thread's SRAM signal is received.

5. (Currently amended) The method of claim 3 wherein the parameter specifies "sdram Swap," the directing will swap out [[a]] the current context and wakes it up when the thread's SDRAM signal is received.

6. (Currently amended) The method of claim 3 wherein the parameter specifies "FBI" ~~will swap~~ swaps out [[a]] the current context and wakes ~~it~~ up the swapped, current context when the thread's FBI signal is received indicating that an FBI CSR, Scratchpad, TFIFO, or RFIFO operation has completed.

7. (Currently amended) The method of claim 3 wherein the parameter specifies "seq_num1_change/seq_num2_change", which swaps out [[a]] the current context and wakes ~~it~~ up the swapped, current context when a value of the sequence number changes.

8. (Currently amended) The method of claim 3 wherein the parameter specifies "inter_thread" which swaps out [[a]] the current context and wakes ~~it~~ up the swapped, current context when the thread's interthread signal is received.

9. (Cancelled)

10. (Currently amended) The method of claim 3 wherein the parameter specifies "auto_push" which swaps out [[a]] the current context and wakes ~~it~~ up the swapped, current context when SRAM transfer read register data has been automatically pushed by a FBus interface.

11. (Currently amended) The method of claim 3 wherein the parameter specifies "start_receive" that swaps out [[a]] the current context and wakes ~~it~~ up the swapped, current context when new data in a receive FIFO is available for this thread to process.

12. (Currently amended) The method of claim 3 wherein the parameter specifies "kill" which prevents [[a]] the current context or thread from executing again until an appropriate enable bit for the thread is set in a CTX_ENABLES register.

13. (Currently amended) The method of claim 3 wherein the parameter specifies "pci" which swaps out [[a]] the current context and wakes ~~it~~ up the swapped, current context when a PCI unit signals that a DMA transfer has been completed.

14. (Previously presented) The method of claim 3 wherein directing further comprises:

an optional_token "defer one" which specifies that one instruction will be executed after this reference before the context is swapped.

15. (Currently amended) A method of operating a multithreaded parallel processor, the method comprising:

~~evaluating a specified parameter~~ receiving an indication of a voluntary swap, the voluntary swap specified in a context-swap instruction ~~to determine a state of an executing context process corresponding to a first thread~~; and

performing, in response to the received indication, a swapping operation to cause an executing context process corresponding to a first thread to be swapped with a different context and associated program counter, corresponding to a different thread that is ready to execute~~, to be selected in accordance with the value of the evaluated specified parameter~~,

wherein the swapped first thread is automatically re-enabled to run at some subsequent context arbitration point.

16. (Previously presented) The method of claim 15 wherein performing swaps a currently running context in a specified microengine to let another context execute in that microengine.

17. (Currently amended) The method of claim 15 ~~wherein the parameter specifies an occurrence of an event~~ further comprising:

receiving a parameter specifying an occurrence of an event; and

waking up the swapped out context in accordance with the parameter.

18. (Currently amended) The method of claim ~~15~~ 17 wherein the parameter specifies "sram Swap", and performing a swapping comprises swapping out [[a]] the current context and waking ~~it~~ up the swapped, current context when the thread's SRAM signal is received.

19. (Currently amended) The method of claim ~~15~~ 17 wherein the parameter specifies "sram Swap", and performing a swapping comprises swapping [[a]] the current context and waking ~~it~~ up the swapped, current context when the thread's SDRAM signal is received.

20. (Currently amended) The method of claim ~~15~~ 17 wherein the parameter specifies "inter_thread" which swaps out [[a]] the current context and wakes ~~it~~ up the swapped, current context when the thread's interthread signal is received.

21. (Original) The method of claim 15 further comprising:

an optional_token "defer one" which specifies that one instruction will be executed after this reference before the context is swapped.

22. (Currently amended) A parallel processor that can execute multiple contexts and that comprises:

a register stack;

a program counter for each executing context;

an arithmetic logic unit coupled to the register stack and a program control store that stores a context swap instruction that causes the processor to:

~~evaluate a parameter~~ receive an indication of a voluntary swap, the voluntary swap specified in the context swap instruction ~~to determine a state of an executing context process corresponding to a first thread~~; and

perform, in response to the received indication, a swap operation to cause [[a]] an executing context process corresponding to a first thread to be swapped with a different context and associated program counter, corresponding to a different thread that is ready to execute, ~~to be selected in accordance with the value of the evaluated specified parameter and which saves an old program counter value~~,

wherein the swapped first thread is automatically re-enabled to run at some subsequent context arbitration point.

23. (Currently amended) The processor of claim 22 wherein the ~~context swap instruction~~ processor wakes up the swapped out context when a specified signal is activated.

24. (Currently amended) A computer program product residing on a computer readable medium for causing a multithreaded parallel processor to perform a function, the computer program product comprising ~~comprises~~ instructions causing the processor to:

~~evaluate a specified parameter to determine a state of an executing context process corresponding to a first thread~~ receive an indication of a voluntary swap, the voluntary swap specified in a context-swap instruction; and

perform, in response to the received indication, a swapping operation to cause an executing context process corresponding to a first thread to be swapped with a different context and associated program counter, corresponding to a different thread that is ready to execute, ~~to be selected in accordance with the value of the evaluated specified parameter~~

wherein the swapped first thread is automatically re-enabled to run at some subsequent context arbitration point.

25. (Currently amended) The product of claim 24 ~~wherein~~ further comprising instructions that cause the processor ~~wakes~~ to wake up the swapped out context when a ~~specified~~ signal corresponding to ~~the~~ a specified parameter is activated, wherein the specified parameter is identifies in a context-swap instruction.